

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE



Membership Publications/Services Standards Conferences Careers/Jobs

**IEEE Xplore**  
 RELEASE 1.0

 Welcome  
 United States Patent and Trademark Office


&gt;&gt; ABSTRACTS

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)
[Quick Links](#)

Welcome to IEEE Xplore

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

[Search Results](#) [PDF FULL-TEXT 1412 KB] [NEXT](#) [DOWNLOAD CITATION](#)

 Request Permissions  
**RIGHTS LINK**

## Transient fault detection via simultaneous multithreading

Reinhardt, S.K. Mukherjee, S.S.

Dept. of Electr. Eng. &amp; Comput. Sci., Michigan Univ., Ann Arbor, MI, USA;

*This paper appears in: Computer Architecture, 2000. Proceedings of the International Symposium on*

Meeting Date: 06/10/2000 - 06/14/2000

Publication Date: 10-14 June 2000

Location: Vancouver, BC Canada

On page(s): 25 - 36

Reference Cited: 21

Number of Pages: vi+328

Inspec Accession Number: 6644850

### Abstract:

Smaller feature sizes, reduced voltage levels, higher transistor counts, and reduced margins make future generations of microprocessors increasingly prone to transient hardware faults. Most commercial fault-tolerant computers use fully replicated components to detect microprocessor faults. The components are lockstepped (cycle synchronized) to ensure that, in each cycle, they perform the same operation with the same inputs, producing the same outputs in the absence of faults. Unfortunately, at a given hardware budget, full replication reduces performance by statically partitioning resources among redundant operations. We demonstrate that a Simultaneously Redundantly Threaded (SRT) processor-derived from a a Simultaneous Multithreading (SMT) processor-provides transient fault coverage with significantly higher performance. An SRT processor provides transient fault coverage by running identical copies of the same program simultaneously as independent threads. An SRT processor provides better performance because it dynamically schedules its hardware resources among redundant copies. However, dynamic scheduling makes it difficult to implement lockstepping, because corresponding instructions from redundant threads may execute in the same cycle or in the same order. This paper makes four contributions to the design of SRT processors. First, we introduce the concept of the sphere of replication, which abstracts both the physical redundancy of a lockstepped system and the redundancy of an SRT processor. This framework aids in identifying the scope of replication coverage and the input and output values requiring special handling. Second, we identify two viable spheres of replication in an SRT processor, and show that one of them provides fault detection while checking only committed stores and uncached loads.

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE


[Membership](#) [Publications/Services](#) [Standards](#) [Conferences](#) [Careers/Jobs](#)
**IEEE Xplore**  
RELEASE 1.0

 Welcome  
 United States Patent and Trademark Office


» ABSTRACT

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)
[Quick Links](#)

Welcome to IEEE Xplore

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

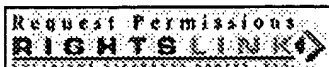
- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

[Search Results](#) [\[PDF-FULL-TEXT 76 KB\]](#) [DOWNLOAD CITATION](#)


## AR-SMT: a microarchitectural approach to fault tolerance in microprocessors

Rotenberg, E.

Dept. of Comput. Sci., Wisconsin Univ., Madison, WI, USA;

*This paper appears in: Fault-Tolerant Computing, 1999. Digest of Papers: Ninth Annual International Symposium on*

Meeting Date: 06/15/1999 - 06/18/1999

Publication Date: 15-18 June 1999

Location: Madison, WI USA

On page(s): 84 - 91

Reference Cited: 24

Number of Pages: xx+357

Inspec Accession Number: 6314443

### Abstract:

This paper speculates that technology trends pose new challenges for fault tolerant microprocessors. Specifically, severely reduced design tolerances implied by high clock rates may result in frequent and arbitrary transient faults. We suggest that fault-tolerant techniques-system-level, gate-level, or component-specific approaches are either too costly for general purpose computing, overly intrusive to the design, or insufficient for covering arbitrary logic faults. An approach in which the microprocessor itself provides fault tolerance is required. We propose a new time redundancy tolerant approach in which a program is duplicated and the two redundant programs simultaneously run on the processor. The technique exploits several significant microarchitectural trends to provide broad coverage of transient faults and redundancy coverage of permanent faults. These trends are simultaneous multithreading, and data flow prediction, and hierarchical processors-all of which are intended to improve performance, but which can be easily leveraged for the specified fault tolerance. The overhead for achieving fault tolerance is low, both in terms of performance and changes to the existing microarchitecture. Detailed simulations of five of the SPEC benchmarks show that executing two redundant programs on the fault-tolerant microarchitecture takes only 10% to 30% longer than running a single version of the program.

### Index Terms:

[computer architecture](#) [fault tolerant computing](#) [redundancy](#) [AR-SMT](#) [fault tolerance](#)

IEEE HOME | SEARCH IEEE | SHOP | WEB ACCOUNT | CONTACT IEEE



Membership Publications/Services Standards Conferences Careers/Jobs

**IEEE Xplore**  
 RELEASE 1.0

 Welcome  
 United States Patent and Trademark Office


» ABSTRACTS

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)

Quick Links

Welcome to IEEE Xplore

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

[Search Results](#) [\[PDF FULL-TEXT 1136 KB\]](#) [DOWNLOAD CITATION](#)

 Request Permissions  
**RIGHTS LINK**

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

## Simultaneous multithreading: Maximizing on-chip parallelism

[Tullsen, D.M.](#) [Eggers, S.J.](#) [Levy, H.M.](#)

Dept. of Comput. Sci. &amp; Eng., Washington Univ., Seattle, WA, USA;

*This paper appears in: Computer Architecture, 1995. Proceedings. 22nd International Symposium on*

Meeting Date: 06/22/1995 - 06/24/1995

Publication Date: 22-24 June 1995

Location: Santa Margherita Ligure Italy

On page(s): 392 - 403

Reference Cited: 34

Inspec Accession Number: 5086803

### Abstract:

This paper examines simultaneous multithreading, a technique permitting independent threads to issue instructions to a superscalar's multiple functional units in a single cycle. We present several models of simultaneous multithreading and compare them with alternative organizations: a wide superscalar, a fine-grain multithreaded processor, and single-chip, multiple-issue multiprocessing architectures. Our results show that both (single-threaded) superscalar and fine-grain multithreaded architectures are limited in their ability to utilize the resources of a wide-issue processor. Simultaneous multithreading has the potential to achieve 4 times the throughput of a superscalar and double that of fine-grain multi-threading. We evaluate several cache configurations made possible by this type of organization and evaluate tradeoffs between them. We show that simultaneous multithreading is an attractive alternative to single-chip multiprocessing; simultaneous multithreaded processors with a variety of organizations outperform corresponding conventional multiprocessors with similar execution resources. While simultaneous multithreading has excellent potential to increase processor utilization, it can add substantial complexity to the design. We examine many of these complexities and evaluate alternative organizations in this space.

### Index Terms:

[cache configurations](#) [computational complexity](#) [computer architecture](#) [fine-grain multi-processor](#) [multiprocessing architectures](#) [multiprocessing systems](#) [on-chip parallelism](#) [maximisation](#) [processor scheduling](#) [processor utilization](#) [simultaneous multithreading](#)